



# Software Requirements Workshop Outbrief

## Co-Leads:

Ceci Albert  
Dr. William Bail  
Lisa Brownsword  
Mr. Blake Ireland  
Dr. Kenneth E. Nidiffer

## Participants:

Erika Chan  
Carl Clavadetscher  
Ed Cutter  
Stanley Hill  
Angela Llamas-Butler  
Tom McGibbon  
Todd McCarty  
Ira Monarch  
Paul Popick  
Ann Reid-Shaw  
Howard Small  
Marion Williams

Department of Defense  
Software in Acquisition Workshop  
October 17, 2007  
Forum to Share Insights, Experiences, Best Practices  
and Recommendations Related to Software in DoD Acquisitions  
NRECA Building, 4301 Wilson Blvd  
Arlington, VA



# Requirements Sub-Panel Approach

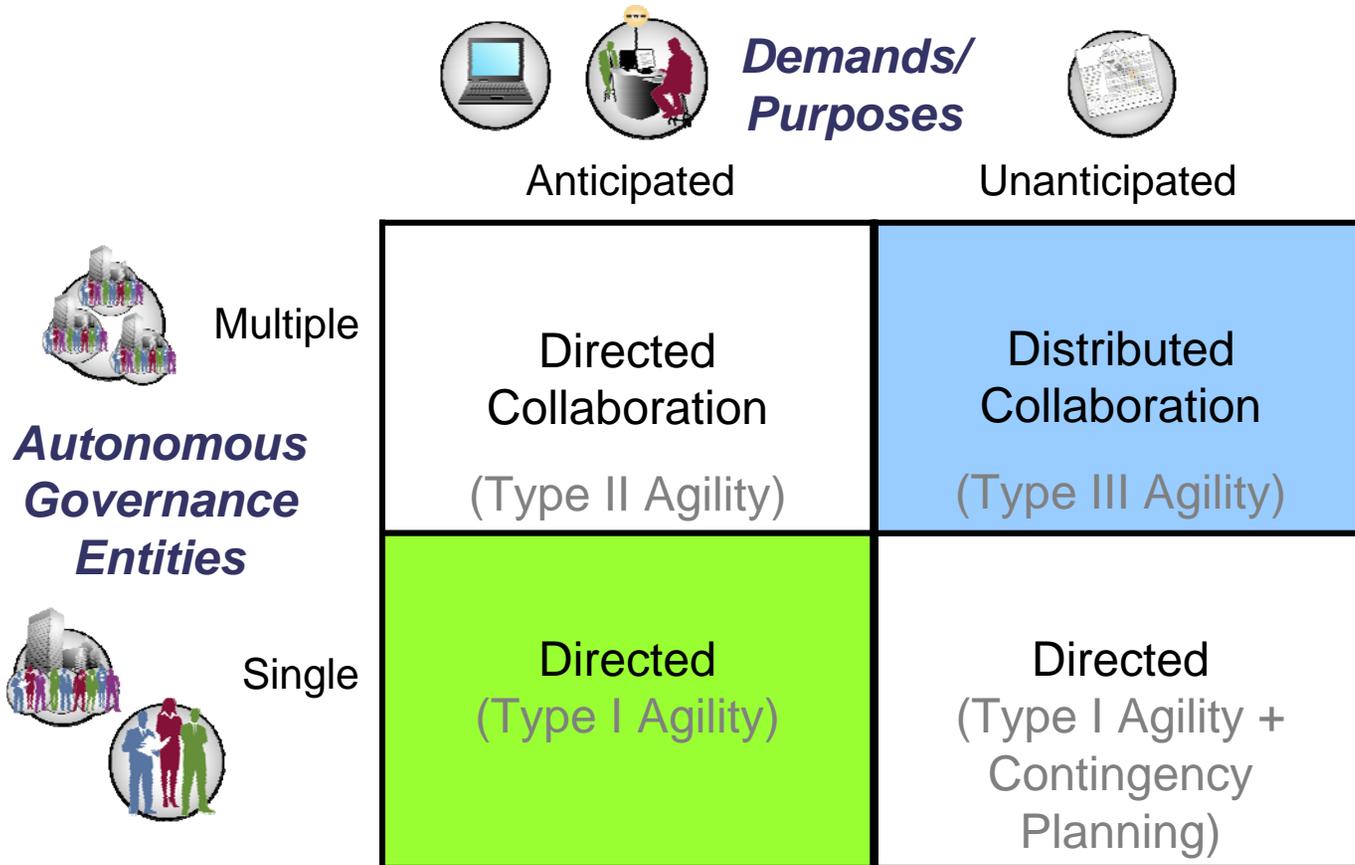
---

- Framed Requirements Management Challenge
  - Leveraged the “Double Challenge in Software Requirements Management” work by Philip Boxer, Lisa Brownsword (today’s co-presenter), Dennis Smith at the SEI.
- Discussed the software requirement issues by “challenge type”
  - Examined the most important boundary conditions
- Identified the key gaps by “challenge type”
  - What are the indicators that this is a real and important issue?
- Developed of actions that should be taken to address the problem
  - Define a specific plan to crystallize concrete progress within the next 6-18 months
  - Define work products and a plan to develop them over 6, 12 and 18 month periods
  - Identify stakeholders relevant to each of these work products





# New Lens Used to Meet Emerging Software Requirements Challenges



Forms of Collaboration from "Architecting Principles for Systems of Systems", by Mark W. Maier  
<http://www.infoed.com/open/papers/systems.htm>



**Software Engineering Institute**

**Carnegie Mellon**

Software in Acquisition Workshop Requirements  
Out brief  
17 October 2007

© 2007 Carnegie Mellon University

# Type 1 - Directed

---



- Characteristic

- One entity has strong development control
- Delivering an end item that is defined in general terms with some level of precision

- Issues and Gaps

- Many problems are the same as those we have been trying to fix for 30 years
- We know what to do however we do not incentivize it, pay for it and train for it (e.g. application of IEEE requirements attributes)
- Lack of early and continuous involvement of all relevant SMEs
- We are making changes to software whose existing behavior we do not understand
- Do not have adequate tools, methods and processes for requirements definition
- How are articulated specifications adequate to development derived from capabilities? – especially in a continuous evolution environment
- Requirements are added, changed or deleted without sufficient engineering
- Component cannot operate alone – no one owns the external relationships





# Type 1 – Directed Recommendations

---

1. Define an effective “software portfolio” management framework
  - Protect the continuity of systems/software and requirement engineering throughout the software life cycle
2. Implement the techniques we know will work and identify any shortcomings
  - Training
  - Incentives
  - Re-examine IEEE tenets for good requirements
3. Find ways to leverage the malleability of software
  - We need new methods to deal with “on the fly” and/or external requirements: Software has the ability to adapt faster than other elements
  - Build and integrate effective modeling existing systems and adding new requirements
  - Identify resources and methods to facilitate planning for extended use
  - Find ways to manage the malleability to minimize risk
4. Change our view/perspective of “sustainment” to “continuous evolution”
  - Codify the processes for “reverse engineering” candidates to extract for reuse – system components from government or industry
  - Look at organizational as well as methods and skills to perform continuous evolution





# Type 3 Distributed-Collaboration

---

## •Characteristic

- No entity has control
- Needed operational capability is often unanticipated

## • Issues and Gaps

- Lots of “potential capability” and no way to unambiguously know what each entity does
  - How do I use what I have (including capability that is in sustainment)
- Capabilities may/will cross portfolios
  - Solutions cross many different “colors” of money
  - Need cross-cut test and configuration
- Urgent need for research for requirements engineering
  - Is testing different?
    - Verification, certification, accreditation
  - How do we specify needed capability without over constraining potential continuous evolution and current capabilities



# Type 3 Distributed-Collaboration Recommendation

---



5. Establish a research program
  - Identify the characteristics of requirements engineering in type III systems and how it is distinguished from type I
  - Start to identify good practices





# Actions For Each Recommendation

---

- Develop a set of white papers (notionally 6-12 mos)
  - Set context and gaps (evaluate existing successes)
  - Identify new ways to bridge gaps – competing ideas
  - Characterize how it is being accomplished
  - How it fits in defense acquisition
  - Vision for tomorrow: Measures of effectiveness
- Conduct a workshop for socializing the new approaches (notionally 12 mos)
  - Test barriers and enablers
  - Develop criteria for pilot
- Find pilot programs to test approaches (notionally 12-18 mos)
- Analyze the results (notionally 18 -24 mos)
- Deploy the approach (e.g. training, guidebooks, policies, etc)

